

Generalized Markup for Literary Texts

DAVID T. BARNARD, CHERYL A. FRASER and GEORGE M. LOGAN
Queen's University Kingston, Ontario, Canada

Abstract

Encoding literary texts for analysis, electronic transmission, or publication requires the marking of various substantive, structural, and formal features. The development of a standard comprehensive markup language for these purposes is a desideratum. The paper offers a set of requirements for such a language, reviews related work, and describes a newly-created standard based on the Standard Generalized Markup Language.

Introduction

When a literary text is encoded in machine-readable form, it is almost always necessary to include more than simply the words that constitute it. This additional information marks certain features of the text which must be distinguishable for the purposes of analysis, for electronic transmission, or for formatting and printing.

Which features are marked depends on the particular applications envisaged and the kinds of textual features that are important in them. For example, analysis may well require that proper names be marked to ensure that capitals that begin a sentence or a line of verse are distinguishable from those that begin a proper name. Punctuation chosen for marking usually includes hyphens at line-ends, periods that indicate ellipses, and those associated with abbreviations. Hyphens occurring at the ends of lines are, of course, marked to distinguish them from those which appear in compound words; periods indicating ellipses or abbreviations are marked to distinguish them from the full stop that ends a sentence.

The marking process just described, focussed on the level of words and sentences, is meant to disambiguate a text. However, unless we wish to view the text, for the purpose of computation, as one long string of characters, textual boundaries—that is, between paragraphs, chapters, and other structural units—must also be distinguishable. Encoding of textual boundaries is also necessary for constructing (or reconstructing) the text in printed form. Similarly, marking any font changes allows reproduction of these typefaces if the text is to be printed. Finally, a particular kind of analysis may require the marking of stylistic features of a text: images, allusions, rhetorical figures, and so on.

There is, though, no recognized standard that offers a taxonomy of textual features, or prescribes how they should be marked. A decade ago, John B. Smith noted the problem posed by this fact, and expressed the hope that “within the next few years a discussion of encoding procedures for natural language texts will seem as antiquated and irrelevant as a discussion of problems

involved in turning spokes for buggy wheels seems today.”¹ This happy situation has not come to pass. Most analytical programs—for example, Smith’s AR-RAS,² or the Oxford Concordance Program³—come with a set of guidelines describing how the text should be prepared, and these are usually different for each program. Thus the lack of a standard normally entails re-marking a text each time a new type of analysis is performed; correspondingly, the emergence of a standard would allow us to move toward a situation in which all texts were prepared electronically according to the standard, and not according to the specific needs of a particular analytic program or research group. This situation would, of course, greatly facilitate the exchange of machine-readable texts between different scholars, different textual archives, and different analytic programs—though the achievement of this last goal would require some rewriting of these programs, or the creation of new ones. Moreover, the existence of a standard would facilitate work with different texts in the *same* archive, for in the present situation the texts held in a single archive typically exhibit a variety of different markup conventions.

The present article introduces a proposed standard, which has been developed by the authors as one facet of a long-term project in humanities computing applications,⁴ involving, among other things, the creation of an archive of texts encoded in standard format. In the interval since writing the article, our sense of the importance and urgency of the problem it addresses has been confirmed by the call by the Association for Computers and the Humanities for a concerted effort to develop a markup standard. One of us (Barnard) is now a member of the ACH’s new Working Committee on Text-Encoding Standards, which is directing the Association’s efforts in this area.

In what follows, we will outline what we take to be the requirements for a standard, review previous work that bears on the creation of one, and describe and briefly illustrate the markup language we have developed.

Requirements for a Standard

Considering the nature and needs of the research community in humanities computing, it seems to us that the requirements for an encoding standard for literary texts are as follows.

1. The standard should be comprehensive: it should provide a way to mark the full range of features that are relevant to literary and linguistic research. At the same time, since complete comprehensiveness can never be achieved, the standard should be extensible, and extensible without great difficulty.
2. The standard should be simple enough to be easily

Correspondence Dr David T. Barnard, Head, Department of Computing and Information Science, Queen's University, Kingston, Ontario, Canada K7L 3N6 (BARNARD@QUCIS.BITNET)

understood by humanistic scholars with limited computer experience and technological sophistication.

3. The standard should be processable by software of only moderate complexity. The development cost of highly complex software renders it unsuitable for all but the most generously funded research; that is, for almost all humanities research.
4. The standard should not be dependent on any particular character set or text-entry device. The scholar should be able to encode documents using existing equipment and should not be required to purchase a specific device.
5. The standard should not be geared to any particular analytic program or printing system. Instead, it should provide a basis for the design of any analytic or publishing application.
6. The standard should describe text in editable form. The ability to revise an encoded text is necessary in order to add information which may have been omitted at the start, and to give the scholar the ability to "tailor" the encoded text to meet specific needs.
7. In order to support collaborative research between scholars at different institutions, the standard should be based on an industry standard for document processing and interchange. This feature would allow the transfer of encoded texts across communication networks without the implementation of special protocols.

Related work

Previous work related to the development of an encoding standard can be broken into two classes. First, a number of researchers and organizations have published guidelines and tips on how to proceed with the task of encoding texts,^{1,5} or described elaborate coding schemes devised for particular purposes.

The most familiar of the sets of guidelines is probably the suggestions for coding included in the manual for the Oxford Concordance Program.³ The user is advised to mark certain punctuation, diacritics, proper nouns, foreign words, font changes, emendations, numerals, and hyphens, as well as textual boundaries. Three different markup *formats* accepted by OCP are described; but OCP does not prescribe a specific set of markup *tags*.

Among the more elaborate coding schemes, we may note two.

- In the Lancaster-Oslo/Bergen (LOB) Corpus of British English,⁶ the texts were encoded using a scheme geared to the needs of linguistic research. Each text in the corpus begins with a tag containing its ordinal number and category, and is followed by an end-of-text marker and a figure representing the total number of words in the text. The encoding scheme includes tags to mark such items as font changes, diacritics, foreign-language materials, abbreviations, paragraphs, and sentences. Grammatical tags were not included in the original coding scheme,

but grammatically tagged versions of both the LOB Corpus and the Brown Corpus have been produced

- "Textcode" is an encoding system developed by researchers at the University of Liverpool and the University of Edinburgh,⁷ to record the grammatical, syntactical, metrical, and accentual features of a text. Tags are entered manually on a special machine-generated marking sheet and then keyed into the computer. The resulting master file resembles the marking sheet, with each line of the sheet corresponding to a record in the master file. Each word of the text is allocated one line on the marking sheet where all the information pertaining to it is recorded. Most information, apart from the name of the text and the word itself, is recorded by numeric entries into single-character columns called "registers."

The developers of Textcode suggest the possibility of adopting the system as a standard for interchange of grammatically marked texts. There are, however, certain drawbacks to Textcode as a standard. Most important, all punctuation has been excluded, rendering the encoding useless for certain types of research. Moreover, Textcode requires that all information be associated with individual words. If one would like to record information about larger structures such as chapters, paragraphs, or sentences, this information must be duplicated for each word in the structure. Finally, the fact that the code is basically numeric constitutes a serious drawback, in that encoded texts are virtually impossible to read without constant reference to the marking-instructions key.

Indeed, while various existing markup schemes are perfectly satisfactory for the purposes for which they were developed, no previous system (or at least any that we know of) provides a satisfactory basis for a standard. For the researcher attempting to create one, their principal value is probably in the help they offer in developing a comprehensive taxonomy of features for which markup tags should be developed, and in the object lessons they offer as to the relative convenience or inconvenience of different approaches to coding.

The second class of relevant previous work is that related to the development of encoding standards for general document processing and transmission. These standards do not explicitly meet the needs of humanistic scholars, but could be used as the basis for one that *does*.

The interchange of documents between different document-processing tools requires a fundamental common understanding of the structure of documents. Several international standards committees, such as the European Computer Manufacturers Association, the International Organization for Standardization, and the International Telegraph and Telephone Consultative Committee, have been working on "Office Document Architecture" and "Office Document Interchange Format" standards to enable the interchange of documents among open systems. The ECMA Standard 101⁸ and the ISO multipart standard on Text Structures⁹ are essentially the same, and allow a document to be interchanged in either final form (for display or printing), processable form (to permit editing and layout revisions), or formatted processable form (which can satisfy both

purposes). The CCITT recommendation T.73¹⁰ is compatible with both the ECMA Standard and the ISO drafts; however, it applies to documents in final form only. IBM's Office Information Architectures¹¹ are a set of specifications for the distribution and management of information (documents, messages, and so on) in an office system network. They define both the form of the information transmitted and the rules governing its use.

For our purposes, though, the most interesting of these standards is the Standard Generalized Markup Language. SGML is the fullest embodiment of the concept of generalized markup, in which all markup is generic rather than process-specific. When a generalized markup language is used, a three-step model of document-processing can be constructed:¹²

1. Recognition: An attribute of the document is recognized—for example, an element with a generic identifier of “paragraph.”
2. Mapping: The attribute is associated with a procedure. The paragraph generic identifier, for example, might be associated, for formatting, with a procedure that skips a line and then indents five spaces.
3. Processing: The procedure is executed.

The advantages of generalized markup are obvious. With process-specific markup, information about a document's attributes is frequently lost. If, for example, formatting requires that book titles and foreign words both be italicized, and an “italic” instruction is inserted to mark both classes of items, it is subsequently impossible for a machine to distinguish between the two classes. In general, generalized markup allows a document to be used in multiple applications without changing the markup.

As its name implies, the Standard Generalized Markup Language *standardizes* the application of generalized markup concepts. SGML is not a markup scheme (that is, it does not prescribe a set of generic identifiers), but rather a methodology for creating such schemes. The language, which is based on IBM's Document Composition Facility Generalized Markup Language, is being developed by the ISO, and has recently been published as an International Standard by that organization.¹² Versions of SGML have already been implemented in several organizations or projects, including British Aerospace,¹³ and the New Oxford English Dictionary Project.¹⁴ The generic markup scheme under development as the Electronic Manuscript Project of the Association of American Publishers^{15,16} is also an SGML implementation. It should be noted, too, that Joan Smith of the National Computing Centre is a strong advocate of SGML.^{17,18}

The features of SGML include the following:¹²

- o An *abstract syntax*, which consists of rules for describing a document's structure and other attributes. These rules apply to all implementations of SGML.
- o A *concrete syntax* that binds the abstract syntax rules to a particular set of characters and quantities. SGML provides a *reference* concrete syntax, but allows users to define alternative concrete syntaxes to meet the needs of specific system environments, devices, or natural languages.

- o *Markup declarations* that permit users to define a specific “vocabulary” of generic identifiers and other attributes that can be employed in specifically defined “document types.”
- o *Entity references*: a system-independent technique for referring to portions of content located outside the document file. For example, a brief entity reference can be substituted for a recurring word or phrase in the document; when the document is processed, the references are resolved into the full word or phrase.
- o *Arbitrary data content*: provision for content not described by the markup language (for example, images, mathematical formulas, etc.).
- o A means of distinguishing processing instructions, which may be entered in certain circumstances, from the descriptive markup.
- o Several optional features, including markup minimization features.

An SGML-based markup language satisfies all the requirements listed earlier for an encoding standard.

1. The requirement of comprehensiveness: The markup language can be as comprehensive as the designers have the foresight to make it. There is no limit to the number of document type definitions or features for which tags can be created. Furthermore, the language is extensible: tags which may have been overlooked can be added easily; if some document has a unique requirement, a new element can be defined to meet this need.
2. The requirement of simplicity: SGML was designed to accommodate to environments in which users interact directly with encoded documents. It therefore provides a simple notation in which delimited strings are inserted directly into the text. (It is the case, though, that the exploitation of all the facilities of SGML can result in a complex textual object.)
3. The requirement that documents be processable by software of moderate complexity: The optional features of SGML can be exploited by powerful software implementations, but less sophisticated systems need not include these features.
4. The requirement that the standard not be dependent on any particular character set or text-entry device. The character set used in a document is specified in the SGML declaration, and any character set that has bit combinations for letters, numerals, spaces, and delimiters will suffice. The variety of keyboards, displays, and text-entry conventions currently in use are accommodated through SGML's optional features and the ability to define an alternative concrete syntax.
5. The requirement that the standard not be geared to any particular analytic program or printing system: Generalized markup is descriptive rather than prescriptive, and need not refer to specific implementation features. Any processing instructions are specially delimited, for easy recognition. As we noted above, SGML is not a markup scheme, but rather a methodology for creating markup schemes. The markup designer may create tags for whatever ele-

ments he or she wishes to identify. The processing associated with the tags is then specified in external procedures.

6. The requirement that the standard should describe text in editable form: Generalized markup languages are intended to describe documents in processable form, and those which exist solely in final formatted form are not covered by SGML.
7. The requirement that the standard allow the interchange of encoded texts across communication networks: The SGML Standard is intended to function at the application layer of the Open Systems Interconnection model. However, the encoded document is in a form suitable for presentation-level interchange, if a transmission header or other envelope structure is added.¹⁹ The Office Document Interchange Format (ODIF) allows an SGML document to be included in its data stream, and could thus serve as the envelope structure. ISO has also considered the possibility of using SGML as an alternative interchange format for documents structured by the Office Document Architecture.²⁰ Moreover, there is a document interchange format specifically designed for SGML.²¹

Results

The details of the markup scheme we have developed are embodied in Fraser's thesis.²² In brief, we have used SGML to describe four document types: prose, poetry, drama, and collections. We have defined over two hundred tags for marking structural entities such as chapters, acts, and paragraphs and the various categories of front and back matter; for disambiguating lexemes such as abbreviations; for facilitating analysis of textual content such as occurrences of particular rhetorical figures; and for recording metrical properties and the properties of keyboarded manuscripts.

The scheme is easy to learn and to apply, provided the markup editor undertakes to learn only those tags which are needed to mark up the document for the desired processing. Tag names have been kept to a few characters, with the shortest names for the most common elements. The "markup minimization" capability afforded by SGML permits tags to be omitted as long as doing so does not create an ambiguity.

Individual scholars are, of course, free to employ the standard in any way they choose. As we pointed out earlier, though, one advantage of *having* a standard is that it facilitates the interchange of encoded texts between scholars and between applications. With this fact in mind, it seemed appropriate to classify the document elements and the corresponding tags into categories, along lines suggested by an assessment of their greater or lesser importance, and their objective or subjective nature. From these considerations, a three-part categorization follows:

1. The first category includes those elements that are of general importance and are objectively determinable. Presumably these elements should be marked in any encoded copy designed to be circulated, or to be deposited in a text archive.

2. The second category includes elements that are of less general importance but are, like those in the first category, objective. A circulated or archived copy need not include these, but its value could only be enhanced if it did.
3. The third category includes all those items which, though they may be of great importance in certain applications, are not objectively determinable: different scholars would be unlikely to agree entirely as to how they should be tagged. Tags in this category should presumably be included in a circulated or archived copy only if the copy includes some warning that they are present.

Example

We offer a brief illustration of the scheme (Figure 1), in the form of a marked-up version of the opening lines of *Hamlet*, together with the *dramatis personae*, as they appear in a standard scholarly edition.²³

This version includes only category 1 tags; in effect, those tags that would be necessary to reproduce a properly formatted version of the passage if it were processed for printing. All these tags are, of course, among those that are specified as legitimate in the document type "drama." Only a few of these tags (though some of the most important ones) happen to occur in this sample. It should be noted that in practical implementations much of the tagging would be done by programs, including the use of function keys for frequently used tags.

The example opens with the document type declaration, which includes an indication that the definition of the type is to be found in a separate file maintained by the computing system, and named "drama def a." Entity declarations follow: these allow the abbreviation of strings (such as names of characters) that appear frequently. When the text is processed, these abbreviations are resolved into the full names. If the user has to create the machine-readable form of the text by keyboarding it (as opposed to optically scanning it or acquiring it from a publisher), the use of such entity references can save many keystrokes.

All tags are enclosed in angle brackets; where an end tag is required, it is identical to the start tag, except that a slash is inserted. The "drdoc" tag, which follows the entity declarations, identifies the work as a play (for the purposes of archival search and retrieval). This tag has several optional attributes, including those specified in the example. Of these, "copytx" indicates that the machine-readable version preserves copy-text lineation; the function of the others is obvious. The "frontm" tag indicates a major structural feature, the beginning of the front matter. (In the example, we have omitted all of this matter except the title page.) The "body" tag marks the end of the front matter and the beginning of the body of the work.

"dpers" indicates the beginning of the *dramatis personae*. In this example we have marked the personae with the "definition list" tag, which identifies a list in which terms are paired with their descriptions. The attributes "tsize" and "compact" give formatting instructions for this list (necessary for applications that involve recon-

```

<!DOCTYPE drdoc SYSTEM "drama.def"
[<ENTITY H "Horatio" >
<ENTITY FO "Fortinbras" >
<ENTITY M "Marcellus" >
<ENTITY FR "Francisco" >
<ENTITY B "Barnardo" > ]>
<drdoc copytx genre='17th century drama' langtx=English
  extent='5 acts, 419 pages'
  subject='A Shakespearean tragedy'>
<frontm>
<title>
<title> Hamlet </title>
</title>
<body>
<dpers> Dramatis Personae
<dl tsize=0 compact>
<dt> Hamlet, <dd> Prince of Denmark
<dt> Claudius, <dd> King of Denmark,
Hamlet's uncle
<dt> The Ghost, <dd> of the late king,
Hamlet's father
<dt> Gertrude, <dd> the Queen,
Hamlet's mother, now wife of Claudius
<dt> Polonius, <dd> councillor of State
<dt> Laertes, <dd> Polonius's son
<dt> Ophelia, <dd> Polonius's daughter
<dt> Horatio, <dd> friend and confidant of Hamlet
<dt> Rosencrantz, <dd> courtier, former
schoolfellow of Hamlet
<dt> Guildenstern, <dd> courtier, former
schoolfellow of Hamlet
<dt> Fortinbras, <dd> Prince of Norway
<dt> Voltemand, <dd> Danish councillor,
ambassador to Norway
<dt> Cornelius, <dd> Danish councillor,
ambassador to Norway
<dt> Marcellus, <dd> member of the King's Guard
<dt> Barnardo, <dd> member of the King's Guard
<dt> Francisco, <dd> member of the King's Guard
<dt> Osric, <dd> a foppish courtier
<dt> Reynaldo, <dd> a servant of Polonius
<dt> Players <dd>

```

Fig. 1

structing a formatted version of the text). Each character name is tagged as a "definition term," and the character's role as a "definition description."

"Act" and "scene" tags have the attribute "num[ber]": in this case, the value of the attribute is 1. The stage direction tag ("stg") has three possible attributes: inl = in-line; ri = flush right; oth = other (those directions that appear at the beginning of a scene). The "speech" tag has two possible attributes: speaker (spk) and audience (aud). Here the value of spk is the name of the speaker. Markup minimization features relieve one of the necessity of including end tags. "&B." and the other items opening with "&" and closing with "." are entity references. The "verse line" tag (vl) has a number of possible attributes: "brk" indicates that the following line is metrically a continuation of the line in which the tag appears.

Future work

The document type definitions described in this article form a basis for further work with standardized document representations. A parser for SGML is required to ensure that type definitions conform to the ISO standard, to ensure that individual documents conform to

```

<dt> A Gentleman <dd> of the court
<dt> A Priest <dd>
<dt> A Grave-digger <dd>
<dt> The grave-digger's Companion <dd>
<dt> A Captain <dd> in &FO 's army
<dt> English Ambassadors <dd>
<dt> Lords, Ladies, Soldiers, Sailors, Messengers, and Attendants
<dd> </dl>
<act num=1> <scene num=1>
<stg oth> Enter BARNARDO and FRANCISCO,
two Sentinels </stg>
<spch spk=Bar> Who's there?
<spch spk=Fran> Nay, answer me Stand and unfold yourself
<spch spk=Bar> Long live the King!
<spch spk=Fran> &B?
<spch spk=Bar> He
<spch spk=Fran> You come most carefully upon your hour
<spch spk=Bar> 'Tis now struck twelve
Get thee to bed, &FR
<spch spk=Fran> For this relief much thanks 'Tis bitter cold,
And I am sick at heart
<spch spk=Bar> Have you had quiet guard?
<spch spk=Fran> Not a mouse stirring
<spch spk=Bar> Well, good night
If you do meet &H and &M,
The rivals of my watch, bid them make haste
<spch spk=Fran> <vl brk> I think I hear them
<stg ri> Enter HORATIO and MARCELLUS </stg>
Stand, ho! Who is there?
<spch spk=Hor> <vl brk> Friends to this ground
<spch spk=Mar> And liegemen to the Dane
<spch spk=Fran> Give you good night
<spch spk=Mar> O, farewell honest soldier, who hath reliev'd you?
<spch spk=Fran>
&B hath my place Give you good night <stg inl> Exit </stg>
<spch spk=Mar> Holla, &B!
<spch spk=Bar> Say, what, is &H. there?
<spch spk=Hor> A piece of him
<spch spk=Bar> Welcome, &H Welcome, good &M
<spch spk=Hor> What, has this thing appear'd again tonight?
<spch spk=Bar> I have seen nothing

```

the type definitions, and to produce a consistent internal representation that replaces tags omitted in markup minimization. We are building a prototype parser at present, which we hope to have in use by the end of 1987.

If the document type definitions are to be useful in the construction of archives, there must be some automated way of producing tagged documents, since keying tags manually into a large collection of texts would be tedious, error-prone, and expensive. We are working on a program that will read the files produced by an optical scanner, infer structural information from the physical layout of the text on the page, and automatically insert major structural tagging.

In addition, we are working on the design of a language for processing the internal representation produced by an SGML parser. Some of the tasks that we want to express in such a language are formatting documents for printing or typesetting (perhaps by mapping into an existing formatting language), extracting portions of document collections specified in a query language, and comparing parsed forms of different versions of the same document.

We will be pleased to respond to requests for more detailed information about the standard, or to receive comments on it.

References

- 1 Smith, J. "Encoding Literary Texts: Some Considerations" *ALLC Bulletin*, 4 (1976), 190-199
- 2 Smith, J. ARRAS [ARchive Retrieval and Analysis System]. Conceptual Tools, 1984.
- 3 Hockey, S., and I. Marriott. *Oxford Concordance Program Version 1.0 Users' Manual* Oxford University Computing Service, 1984.
- 4 Barnard, D., R. Crawford, and G. Logan "Working Paper 1: Project Outline." Project Mnemosyne, Queen's University, 1984.
- 5 Preston, M., and S. Coleman. "Some Considerations Concerning Encoding and ConCORDing Texts" *Computers and the Humanities*, 12 (1978), 3-12
- 6 Johansson, S., G. Leech, and H. Goodluck *Manual of Information for the Lancaster-Oslo/Bergen Corpus of British English*. Department of English, University of Oslo, 1978
- 7 Cairns, F., P. Craven, and J. Howie. "Textcode Grammatical, Syntactical, Metrical, and Accentual Information in Machine-Readable Form" *ALLC Bulletin*, 9, No 3 (1981), 13-18.
- 8 *Office Document Architecture* (final draft revised) EC-MA/TC29/85/16, April 1985.
- 9 ISO/DIS 8613/1-6. *Information Processing—Text and Office Systems—Office Document Architecture (ODA) and Interchange Format*. Geneva, 1986.
- 10 *Document Interchange Protocol for the Telematic Services*. CCITT Recommendation T.73, 1984.
- 11 International Business Machines Corporation. *Office Information Architectures Concepts*. Document Number GC23-0765, 1983.
- 12 ISO 8879. *Information Processing—Text and Office Systems—Standard Generalized Markup Language (SGML)* Geneva, 1986
- 13 Clarkson, G. "SGML in Technical Publications at British Aerospace." A paper presented to the European chapter of the SGML Users' Group at IBM Southbank, London, September 1985.
- 14 Stubbs, J., and F. Tompa. "Waterloo and the New Oxford English Dictionary Project." A paper presented to the Twentieth Annual Conference on Editorial Problems, University of Toronto, November 1984
- 15 Association of American Publishers "Project Plan and RFP for Electronic Manuscript Project." *EMP Document #1*, May 1983.
- 16 Davis, B. "Electronic Manuscript Preparation and Markup" *Bulletin of the American Society for Information Science*, December/January 1987.
- 17 Smith, J. "The Standard Generalized Markup Language for Data Base Publishing." A paper presented to the 7th International Conference on Computers and the Humanities, Provo, Utah, June 1985
- 18 Smith, J. Reports from Chairmen of Specialist Groups Standards. *Literary and Linguistic Computing*, 1, No. 3 (1986), 191-192 And see now "The Standard Generalized Markup Language (SGML) for Humanities Publishing", *Literary & Linguistic Computing*, 2, No. 3 (1987), 171-175
- 19 Mason, J. "Architectural Relationships between SGML and ODA/ODIF." ISO Individual Contribution, January 1984.
- 20 "Use of SGML in the Context of ODA/ODIF." Proposed Statement of Intention. WG3 and WG5 joint meeting, Toronto, November 1984
- 21 ISO/DIS 9069 *Information Processing—SGML Support Facilities—SGML Document Interchange Format (SDIF)*. International Organization for Standardisation Geneva, 1986.
- 22 Fraser, C. *An Encoding Standard for Literary Documents* M.Sc. Thesis, Department of Computing and Information Science, Queen's University, May 1986
- 23 Shakespeare, William *Hamlet*. Ed Harold Jenkins. The Arden Edition of the Works of William Shakespeare Methuen, 1982